SIDLEY AUSTIN LLP
SIDLEY

SIDLEY AUSTIN LLP
555 WEST FIFTH STREET
LOS ANGELES, CA 90013
(213) 896 6000
(213) 896 6600 FAX

BEIJING HONG KONG SHANGHAI
BRUSSELS LONDON SINGAPORE
CHICAGO LOS ANGELES TOKYO
DALLAS NEW YORK WASHINGTON, DC
GENEVA SAN FRANCISCO

FOUNDED 1866

FACSIMILE/TELECOPIER TRANSMISSION

From:	**Name:**	James E. Wright II
	Voice:	(213) 896-6094
To:	**Name:**	Board of Patent Appeals
	Company:	U.S. Patent and Trademark Office
	Facsimile#:	571-273-8300
	Voice Phone:	
	Subject:	Appeal Brief, Application Serial No. 09/757,849

Date: 2/20/2006 **Time**: 7:25:03 PM **No. Pages (Including Cover)**: 50

Message:

Please find attached an Appeal Brief and related correspondence with respect to Application Serial No. 09/757,849.

* * * * * * * * * * *

* * * * * * * * * * *

Problems with this transmission should be reported to:

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	James E. Wright	Examiner:	Shahid Al Alam
Serial No.:	09/757,849	Art Unit:	2172
Filing Date:	January 9, 2001		
Title:	SYSTEM FOR SEARCHING COLLECTIONS OF LINKED OBJECTS		

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being sent to the United States Patent and Trademark Office by facsimile transmission to 571-273-8300 on February 20, 2006.

James E. Wright

Sir:

In response to the Office communication of January 20, 2006, regarding the above numbered Application, and per the helpful instructions of the Examiner during our conversations in the interim, please find attached a new copy of the brief first submitted on April 30, 2004 with a revised Summary of the Claimed Subject Matter, section headings and appendices. However, it is noted that this appeal brief (and the rejected brief filed June 1, 2005 and referred to in the January 20 Office communication) is being filed pro se. The Applicant thus respectfully submits that the ground of rejection offered for the brief in the January 20 Office communication therefore was, and is, inapplicable.

37 CFR § 41.37(c)(1) states that "*a brief filed by an appellant who is not represented by a registered practitioner need only substantially comply with paragraphs (c)(1)(i) through (c)(1)(iv) and (c)(1)(vii) through (c)(1)(x) of this section*". The express basis of both (a) the checked rejection (#4) of the January 20 Office communication, and (b) the requirement that the brief contain a Summary of the Claimed Subject Matter, is 37 CFR § 41.37(c)(1)(v), but by the terms of § 41.37(c)(1), § 41.37(c)(1)(v) is inapplicable to this brief because it is being filed pro se. The Applicant therefore respectfully submits that the rejection of the appeal brief for alleged failures in the Summary of the Claimed Subject Matter was unnecessary in accordance with the applicable regulation. The Applicant further respectfully submits that the secondary ground for rejection set forth in the "Other" section of the January 20 Office communication, regarding the section's title, is not necessary either, pursuant, *e.g.*, to MPEP § 1205.03, which states that "*The examiner should not require a corrected brief for minor non-compliance in an appeal brief (e.g., the brief has a minor error in the title of a section heading).*"

Finally, the Applicant respectfully directs the Office's attention to the fact that this Appeal and Brief have now been pending for over 21 months without substantive response. Therefore, in order to expedite the resolution of the issues on appeal, the Applicant respectfully requests an immediate transfer to the Board of Patent Appeals for this Appeal and Brief. If there is an additional appeal brief fee beyond the fee already charged in respect of the original submission, please charge it using the attached credit card charge authorization form, however, in light of the fact that the rejections causing resubmission were not in conformance with the applicable regulations, and the appeal brief fee for this appeal has already been paid, the Applicant respectfully submits that no new charge for this resubmission is warranted or should be required.

Respectfully submitted,

James E. Wright
President, CommSoft Corporation
Date: 2/20/06

James E. Wright
CommSoft Corporation
P.O. Box 712280
Los Angeles, CA 90071
(818) 632-7135

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	James E. Wright	Examiner:	Shahid Al Alam
Serial No.:	09/757,849	Art Unit:	2172
Filing Date:	January 9, 2001		
Title:	SYSTEM FOR SEARCHING COLLECTIONS OF LINKED OBJECTS		

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being sent to the United States Patent and Trademark Office by facimilie transmission to 571-273-8300 on February 20, 2006.

James E. Wright

Sir:

APPELLANT'S BRIEF UNDER 37 CFR § 1.192

Real Party in Interest

The real party in interest in the present application is CommSoft Corporation, by virtue of an assignment by the inventor, James Wright, recorded at Reel 011850, Frame 0153.

Related Appeals and Interferences

There are no related appeals or interferences.

Status of Claims

Claims 1-17, 19, and 20 are pending in the application. Claim 18 was cancelled by the Office Action Response filed February 26, 2003.

Status of Amendments

There are no unentered amendments.

Summary of the Claimed Subject Matter

The present invention includes methods of searching collections of linked objects and displaying the results. The invention is of particular utility when the links between objects (*e.g.*, legal citations or bibliographic references) themselves tend to convey useful information about the objects or their relevance to the search.

According to the inventive methods of independent Claims 1 and 2, once a search group is acquired and a set of links therefrom determined (as discussed, *e.g.*, in paragraphs 1 – 3 of the Detailed Description, on numbered page 2 of the July 11, 2002 published application, and as depicted, *e.g.*, in Figure 2), the links from the target objects are used to determine at least one display attribute of the search set when it is displayed to the user. For example, links may be displayed as arrows or other connectors on a graph (as required by independent Claim 16 and shown, *e.g.*, in Figure 2), or otherwise, or color, shape, size, position, highlighting, graphical flags, and/or labeling text may be used to convey information about the links (as discussed, *e.g.*, in paragraph 4 of the Detailed Description, on page 2 of the published application, and shown, *e.g.*, in Figure 5).

The objects and links displayed may furthermore be layered as additionally required by independent Claim 2 (as shown, *e.g.*, in Figures 2, 3, and 4, and discussed, *e.g.*, in paragraphs 4 – 8 of the Detailed Description, on pages 2 and 3 of the published application). The objects may also be annotated with user notes or other information as required by independent Claim 16 (as discussed, *e.g.*, in paragraphs 4 and 6 of the Detailed Description, on pages 2 and 3 of the published application).

Issues

There are two issues presented:

1. Whether claims 1, 16, 17, 19, and 20 are anticipated by Shah, *et al.*, "Infoharness: Managing Distributed, Heterogeneous Information" *IEEE Internet Computing*, Nov/Dec 1999 (hereinafter, "Shah").

2. Whether claims 2-15 are obvious over Shah in view of U.S. Patent No. 5,983,267 to Shklar, *et al.* (hereinafter, "Shklar").

Grouping of Claims

The claims do not stand or fall together, for the reasons set forth in the following section.

Argument

The Prior Art

Shah describes a metadatabase system for indexing heterogeneous groups of documents and images. The system allows searching of metadata to locate documents, and allows a user to create associations between objects. For example, a user may define a group of documents that all relate to the same subject. Shah calls the creation of such associations is called "annotation."

Shklar describes another system for representing data of heterogeneous types. The system tries to provide a uniform presentation format for the heterogeneous data by analyzing its internal organization (*e.g.*, into paragraphs, sections, articles, chapters, or frames), and displaying selected portions of the data.

Claim 1

Independent claim 1 recites a method of searching a collection of linked objects and displaying the results. According to the method, a search group of heterogeneously typed objects is acquired (for example, by keyword searching or other known techniques). The objects are characterized in that at least one of the searched objects comprises a link to another object (for example, a bibliographic citation from one paper to another). For at least a portion of the objects in the acquired search group, the internal links are "followed" to determine their target objects. This determination includes determining whether the target objects are themselves inside or outside of the search group. Finally, at least one member of the acquired search group is displayed, where some aspect of the display (a "display attribute") is determined by the set of target objects. (For example, an object may be color-coded to indicate that it contains links to other objects within the search group).

Claim 1 currently stands rejected as anticipated by Shah. However, it contains two features not found in or suggested by Shah: *determination of link targets* and *using the set of linked targets to determine a display attribute.* While Shah allows a user to view a document retrieved from a search, neither user nor search engine necessarily follows any embedded links in that document to determine their targets. The mere loading of an object containing a link as

suggested by Shah clearly does not determine the identity of the *link target* as required by the claim. A single target may have multiple incoming links, for example through different domains by reference to the examiner's HTML reference, and likewise, links may be either relative or absolute, or dynamic, such that the same apparent link "address" from two different objects or even from the same object at different times might yield two different link targets.

Further, and critically, there is absolutely no suggestion that either search engine or user should determine whether link targets are inside or outside of the originally acquired search group, as recited by claim 1. In fact, failing to probe the target objects during the search to determine their identities, there is no way to even determine whether a link target would be inside the original search result group, since the link address from one object (e.g., an HTML page) to another might not be the same "address" followed to reach the second object as it was located in the original search.

In addition, there is no disclosure in Shah of using the set of targets linked to by objects in a search group to determine display attributes of the group. The Examiner has suggested that this feature is satisfied simply by displaying an HTML document with underlined links. However, again, in this case, the display attribute (underlining) is not determined by the *targets* of the links, which are not loaded or searched by the browser, and in fact may not even exist (e.g. dead or broken links), but solely by the original HTML page and the browser settings. The undetermined *link targets* therefore cannot possibly be the basis for changing the display attributes of the group or in this example the originally displayed object (the HTML page).

Since claim 1 contains two features not found in Shah, it cannot be anticipated under 35 U.S.C. § 102 by that reference. In addition, it is not obvious over Shah, because these two features are not suggested or in any way obvious modifications of that reference.

Claim 2

Independent claim 2 includes all of the features of claim 1, and also recites that displayed representation of search objects are arranged in layers, which may be independently hidden or shown by a user. This claim currently stands rejected as obvious over Shah in view of Shklar. The arguments set forth above in connection with claim 1 and Shah apply equally to claim 2. Nothing in Shah renders the features of *determination of link targets* and *using the set of linked*

targets to determine a display attribute as obvious. These deficiencies are not remedied by Shklar, which is relied upon primarily to teach the use of layers.

To the extent that Shklar discusses links at all, it describes internal links within a single document, linking one section of the document to another. In contrast, claim 2, like claim 1, recites that an object comprises a link to *another* object – an "external" link. Thus, the links in Shklar are not even the same type as those recited in claim 2. Further, nothing in Shklar suggests determining link targets as recited in claim 2. Not only is there no disclosure of determining whether link targets are inside or outside of the originally acquired search group, such a determination would not even make sense in the context of internal links (since by definition, they point to an object in the search group – the same object from which they originate). Similarly, the set of target objects is not used to determine a display attribute, because the set of target objects is always the same for a document with only internal links – it consists solely of the originating object.

Since neither Shah nor Shklar suggests determining whether linked objects are inside or outside the search group, or using the set of linked objects to determine a display attribute, they cannot render claim 1 or claim 2 obvious under 35 U.S.C. § 103.

Claims 3-15

Claims 3-15 depend (directly or indirectly) in the alternative from either claim 1 or claim 2, and thus cannot be considered obvious over the combination of Shah and Shklar for the reasons set forth above. However, certain of these claims also recite additional features not found in either reference, and thus must independently be considered nonobvious.

Claim 4

Claim 4 recites that representations of a plurality of objects from a search set are displayed on a graph. The specification defines a graph at page 4, lines 11-13, as "a two-dimensional or three-dimensional visual representation of linked objects, where a link is displayed as a connector." An example of representations of objects displayed on a graph may be found in Figure 2 of the present application. Nothing in either Shah or Shklar suggests showing representations of linked objects on a graph. The Examiner has cited page 22, right column, lines 3-23 of Shah for this proposition, but that reference teaches only display of a single

object itself, rather than a display of representations of multiple related objects on a graph. No connectors are shown or described, and only the single object is shown.

Claim 5

Claim 5 recites that representations of a plurality of objects are displayed, and that at least one link between objects is depicted by a connector between these representations. As discussed above, neither Shah nor Shklar teaches or suggests displaying a plurality of object representations and a connector between representations.

Claim 6

Claim 6 depends from claim 5, and further adds that the connector has a display attribute determined by (i) the source object type, (ii) the target object type, or (iii) the link type. Shah and Shklar do not even show connectors, and certainly do not display those connectors differently depending on type. Further, neither reference makes any suggestion that links may have different types.

Claim 8

Claim 8 recites that determining link targets includes not only determining the set of all objects linked to by the original search group, it further includes recursively determining the set of all objects linked to by the expanded set of objects consisting of the union of the search group and the set of objects linked to by the search group. Shah and Shklar do not determine the set of objects linked to by a search group, and certainly do not determine "second-order" links by following a chain of outward links.

Claims 16, 17, 19, and 20

Independent claim 16 recites a search method that includes annotating the results of a search. That is, a user may attach notes to individual objects returned by the search, and these notes may be selectively displayed when viewing the search results. The search results are displayed on a graph. This claim (and its dependent claims 17, 19, and 20) currently stands rejected as anticipated by Shah.

As discussed above, Shah does not teach or suggest the display of representations of search objects on a graph, and for that reason alone, cannot anticipate claims 16, 17, 19, and 20.

In addition, while Shah does use the term "annotate," it does not do so in the same sense as the present application, and cannot be considered to describe this feature of claim 16. In Shah, "annotation" is not the attaching of a note to a specific object, but is the classification of an object into a group by establishing an "annotation relationship" between objects. Such a relationship cannot anticipate the attachment of a note to a single object. Further, Shah does not describe the selective display of annotations, as also recited by claim 16.

Claim 20

In addition, dependent claim 20 adds that representations of objects are displayed with connectors representing links between the objects, and that these links may be annotated. As discussed above, Shah does not teach the display of connectors to represent links between objects, and certainly does not teach the annotation of such links. Thus, claim 20 is also not anticipated for this independent reason.

Conclusion

For the reasons set forth above, claims 1, 16, 17, 19, and 20 are not anticipated by Shah, and claims 2-15 are not obvious over Shah in view of Shklar. Applicant therefore respecfully requests that the Board remove all rejections and allow the present claims to pass to issuance.

If there is an additional appeal brief fee beyond the fee already charged in respect of the original brief, please charge it using the attached credit card charge authorization form.

Respectfully submitted,

James E. Wright
President, CommSoft Corporation
Date: 2/20/06

James E. Wright
CommSoft Corporation
P.O. Box 712280
Los Angeles, CA 90071
(818) 632-7135

CLAIMS APPENDIX

1. (Original) A method of searching a collection of linked objects and displaying the results, comprising:

acquiring a search group of heterogeneously typed objects, wherein at least one of the objects comprises a link to another object;

determining for at least a portion of the objects in the search group a set of targets of links from the objects, including determining whether the link targets are inside the search group; and

displaying a representation of at least one searched object, the representation having at least one display attribute determined by the set of link targets.

2. (Original) A method of searching a collection of linked objects and displaying the results, comprising:

acquiring a search group of objects, wherein at least one of the objects comprises a link to another object;

determining for at least a portion of the objects in the search group a set of targets of links from the objects, including determining whether the link targets are inside the search group; and

displaying a representation of at least one searched object, the representation having at least one display attribute determined by the set of link targets,

wherein displayed representations are arranged into a plurality of display layers, and wherein the display layers can be independently hidden or displayed.

3. (Original) The method of claim 1 or 2, wherein the display attribute is selected from the group consisting of color, shape, size, position, highlighting, graphical flags, and labeling text.

4. (Original) The method of claim 1 or 2, wherein representations of a plurality of objects are displayed on a graph.

5. (Original) The method of claim 1 or 2, wherein representations of a plurality of objects are displayed, and wherein at least one link between objects is depicted by a connector between the representations.

6. (Original) The method of claim 5, wherein a display attribute of the connector is determined by a property selected from the group consisting of the type of the linking object, the type of the link target, and the type of the link.

7. (Original) The method of claim 1 or 2, wherein a display attribute of the representation is determined by object metadata.

8. (Original) The method of claim 1 or 2, wherein determining link targets includes recursively determining targets of links of an expanded set of objects comprising the original search group and the objects linked to by the search group.

9. (Original) The method of claim 8, wherein the recursion level is in the range of 1-10.

10. (Original) The method of claim 1 or 2, wherein the search objects comprise documents selected from the group consisting of legal opinions, legal treatises, statutes, briefs, and law review articles.

11. (Original) The method of claim 1 or 2, wherein the search objects comprise scientific or medical writings, and wherein the links comprise citations to other scientific or medical writings.

12. (Original) The method of claim 1 or 2, wherein the search objects comprise patents and patent applications and the links comprise references to related patents and patent applications.

13. (Original) The method of claim 1 or 2, further comprising annotating at least one of the search objects.

14. (Original) The method of claim 1 or 2, wherein at least a portion of the searched objects and link targets are classified into a plurality of groups, further comprising setting a display attribute for all members of a group.

15. (Original) The method of claim 1 or 2, wherein displayed representations are sorted on at least one axis according to a property of the objects represented.

16. (Previously presented) A method of searching a collection of objects and displaying the results, comprising:
acquiring a first search group of objects;
displaying a representation of at least a portion of the first search group of objects on a graph; and
annotating one or more members of the first search group of objects, wherein annotations may be selectively displayed with the representation of the annotated objects.

17. (Original) The method of claim 16, further comprising:
acquiring a second search group of objects; and
displaying a representation of at least a portion of the second search group of objects, wherein displaying the representation of annotated objects that are members of both the first search group and the second search group includes selectively displaying annotations of the objects.

18. (Cancelled)

19. (Original) The method of claim 16, wherein the objects include links to other objects, and wherein at least a portion of the links are displayed as connectors between representations of the objects.

20. (Original) The method of claim 19, further comprising annotating one or more links.

EVIDENCE APPENDIX

Each of the following items was entered in the record by the Examiner's Office Action of November 1, 2002:

Item 1: Shah, *et al.*, "Infoharness: Managing Distributed, Heterogeneous Information", *IEEE Internet Computing*, Nov/Dec 1999.

Item 2: U.S. Patent No. 5,983,267 to Shklar, *et al.*

INFOHARNESS: Managing Distributed, Heterogeneous Information

Using metadata extraction methods, InfoHarness provides integrated, rapid access to huge amounts of heterogeneous information, regardless of type, representation, location, and medium.

KSHITIJ SHAH AND AMIT SHETH
University of Georgia, Large-Scale Distributed Information Systems Lab

Today, important information is scattered in so many places, formats, and media, that getting the right information at the right time and place is an extremely difficult task. Developing a single software product, for example, includes the creation of documents ranging from the requirements specification and project schedules to marketing presentations, multimedia tutorials, and more. Each document may be created by a different person using a different tool, and each may be stored in a different place.

InfoHarness is an information integration system, platform, and tool set that addresses these problems, managing huge amounts of heterogeneous information in a distributed environment. Through a powerful, consistent user interface, InfoHarness provides rapid search of and access to information assets including documents and parts of documents, mail messages, images, code files, video clips, Web pages with URLs, InfoHarness queries, and views of relational tables. The system makes all these artifacts available without relocating, restructuring, or reformatting the data.

Instead, InfoHarness associates each original artifact with an extensible set of metadata—for example, the artifact's type, location, access rights, owner, and creation date. Using the metadata, the system rapidly, and largely automatically, creates information repositories accessible through any HTTP-compliant browser. Users can browse or query a repository for items of interest.

In this article, we explain the InfoHarness approach to metadata extraction and heterogeneous information management. We also describe VisualHarness, which extends the basic system to accommodate visual data. Finally, we describe how to use InfoHarness tools and hooks to create other Web-based, information-intensive applications.

 1089-7801/99/$10.00 ©1999 IEEE

METADATA-CENTERED APPROACH

Other researchers have investigated the use of metadata to support runtime access to original information[1] and data warehousing and mining for automatic metadata extraction.[2] In building InfoHarness, we refined and synthesized these ideas to provide advanced search and browsing capabilities without imposing constraints on information suppliers or creators. Moreover, users can logically model the information space to best fit their needs.

Our goals with InfoHarness were to

- provide integrated access to a networked heterogeneous information source without forcing information relocation or reformatting,
- create and manage metadata for easy retrieval and decision support,
- categorize related information items into collections to provide a logical information organization,
- allow scalable searches,
- create and manage relationships among groups of information items without affecting the information artifact contents,
- perform programmatic actions on retrieved information, and
- make information access and dissemination easy, low-cost, and ubiquitous.

We translated these high-level requirements into a scalable, extensible architecture.

Metadata Classification

We classify pieces of metadata by how successfully they capture the data and information content of documents from various media types. Metadata can be content-dependent, content-descriptive (a special case of content dependence), or content-independent. In modeling application domain-specific information, it is crucial to capture the semantic content at a level of abstraction similar to that a human would employ.

Content-dependent metadata depend only on the original data's content. It is easy to process text to identify such metadata, usually represented as keywords, but for visual information it is very hard to extract. When this kind of metadata is not extracted automatically from the content itself, we call it content-descriptive. Content-descriptive metadata come from an analysis of the content. When this is not possible, they are derived intellectually. An example would be to identify that a flower in an image has a "sweet and rosy" fragrance. Examples of content-descriptive metadata are the document vectors in Latent Semantic Indexing[3] and the complete inverted Wide Area Information Services index[4]—these list the frequency and position of text units in a document.

Content-descriptive metadata can be domain-dependent or -independent. Identifying that a particular shape of an object in an image is that of a particular type of an airplane, for example, is domain-dependent metadata. A description of the structure of a multimedia document is domain-independent metadata.

InfoHarness is a metadata management system with a generic metadata storage system as its metabase.

Creating content-independent metadata is like attaching a tag to the data regardless of its contents. Examples are a document's creation date and location.

InfoHarness Metadata Infrastructure

InfoHarness is basically a metadata management system with a generic metadata storage system as its metabase. The system extracts metadata from information artifacts and then creates metadata objects that represent the original artifacts with related attributes.

The system uses these metadata attributes to perform various tasks. For example, InfoHarness tools could use a keyword list associated with an artifact—a content-dependent attribute—to build a keyword index. In addition to full-text and keyword searches, InfoHarness lets users perform searches on metadata attributes. These prove especially useful when the user has some knowledge about the information artifacts' metadata semantics.

The system's metabase consists of InfoHarness objects that encapsulate the physical data represented in the metabase. Each IHO can have any number of metadata attributes. The InfoHarness server uses the metabase at runtime to build the user interface, browsing structure, and so forth.

Metadata Extraction and Management

InfoHarness can preprocess information artifacts to generate metadata, or it can extract the metadata at runtime. The system's extractors automatically generate metadata based on the media type. For example, a text extractor filters out relevant words and indexes them. Metadata extractors for dates and subjects generated from mail messages return lines starting with the keywords *date* and *subject*. C or C++ extractors recognize logical constructs such as functions, classes, and subclasses.

Extracting domain- and content-dependent metadata from images would involve anticipating the range of user queries and is not feasible except for cases when a domain is highly controlled and well understood. Instead, one could extract domain-independent but content-dependent metadata such as color and shape during preprocessing and other information such as patterns and outlines at access time.[5] Content-independent metadata such as size and location are of course available during preprocessing. Metadata such as container hierarchies for multimedia can either be extracted or explicitly supplied by the metabase designer. We are currently working on automatic or semiautomatic generation of domain-dependent metadata, which helps in associating semantics with the contents.

InfoHarness provides a metabase infrastructure for Web-based information insensitive applications.

Extracting any type of metadata is dependent on the range of user queries. For example, a query might use the size of the data as a retrieval criterion when transport costs are important. Also, metadata could control the presentation and dynamic composition of retrieved information.

An InfoHarness prototype used Illustra to investigate how an object-relational database could be more effective for querying the metadata itself and letting the user browse stored metadata prior to building queries. Systems set up this way also let users manipulate metadata in different media types intuitively. For example, we might use metadata to relate unstructured data such as images with their structured data representations. In such cases it would be advantageous to store the metadata as annotations to the original information.

If InfoHarness stores the metadata this way, it can easily modify the metadata to reflect changes in the information contents or the content descriptions—location, for example. Such systems can also encapsulate type-specific functions along with the metadata to allow associative searches for unstructured information by using the metadata representing its features. An example would be to associate an image-processing function for land cover identification with a certain type of satellite map. The content-descriptive metadata that the system stores—container hierarchies for a multimedia object, for example—determine how the user browses the information.

Another issue in metadata storage is the metadata's location for remote queries. The system can store metadata locally or at remote sites, or it can prefetch the metadata for a query and then use it to intelligently analyze the query. For example, if the system determines from the metadata that the querying site cannot handle the size of the results, it can take appropriate action rather than retrieving the information and then failing.

Logical Structuring and Browsing

The InfoHarness metabase has an object layer over it, which it manages as a relational database. This metabase represents real-world information artifacts as encapsulated metadata IHOs, and it includes constructs that allow arbitrary typed relationships between IHOs. The IHOs can be simple, directly encapsulating real-world information artifacts, or abstract, representing existing InfoHarness structures. One such abstract IHO type is *collection*, a logical grouping that the InfoHarness server interprets to provide a browsing and searching structure. That is, the browsing structure with which the user navigates the InfoHarness repository is equivalent to the collection hierarchy. The metabase is available to multiple InfoHarness servers and remains persistent between their multiple instances.

Because the metabase uses a traditional database management system, it can handle concurrent updates and can be dynamically extended and modified while the InfoHarness servers are running. This dynamic, persistent metabase has a clean interface to the server on one end and the InfoHarness admin-

istrative tools on the other. However, as extractors return objects and InfoHarness stores them in a relational database, the system faces the limitations and complexities of any object-relational mapping.

LOGICAL METADATA-BASED INFORMATION SPACE STRUCTURING

Artifacts become available through InfoHarness after the system registers them. During registration, the system extracts metadata from the artifact and creates an IHO encapsulating various pieces of metadata about it. The IHO serves as a handle for the actual artifact. Each IHO contains several attributed metadata—some mandatory and maintained by the system, such as document type, object ID, and artifact location. There is no fixed schema of attribute names in InfoHarness, so application builders can create new attributes as needed without modifying the code.

Each IHO has a particular object type attribute indicating the representation or format of the bytes making up the artifact's information and the semantics interpreting this representation. For example, a document type called Bellcore-FrameMaker-Document might have a FrameMaker binary storage format and semantics designated by one of the Bellcore standard Frame templates. InfoHarness can handle any document type that represents a class of documents. The document type's specification declares important information such as how to extract metadata and text from an artifact and what legal actions a user can perform on an IHO.

Each document type has an extractor that pulls out a certain fixed set of metadata attributes. For example, the HTML extractor pulls out a document title, while a Frame extractor pulls out additional values based on FrameMaker variables. The extractors can be any script or executable on the system, but they must be wrapped to conform to the standard administrative interface. For example, a thin Perl script could wrap an existing troff extractor, reformatting the extractor's output into the InfoHarness message exchange format.

We mentioned earlier that there is no fixed schema over an entire InfoHarness repository. In addition, there is no fixed schema for a single document type. Different documents of the same type might have different sets of attributes, none of which are ever enumerated in any document type schema. This flexibility lets an application builder create applications that use new metadata attributes without having to modify system schemas.

When InfoHarness registers an artifact, it stores the artifact's IHO in a repository. Most sites have a single repository holding all IHOs, but some create multiple repositories for large-scale partitions of their information space. Two groups sharing the same InfoHarness software installation might create a repository for each project. Each installation has a single metabase storing all metadata.

InfoHarness uses collections as modeling constructs to organize the IHOs in a repository's information space. A collection represents a set of other related IHOs. For example, a user could create a collection containing all design documents for a particular software package release. Collections can also contain other collections; this nesting forms a repository's collection graph. The collection model consists of sets of relationship structures imposed upon a repository rather than containment structures in which to place and store objects. Collections do not encapsulate information artifacts but participate in relationships with other IHOs or collections; this lets the user build arbitrary logical models. A collection-membership relationship exists between each collection object and member object. InfoHarness understands this kind of relationship and can interpret it during browsing and searching. The application-building tools could interpret these relationships in any context a designer desires.

VisualHarness extends InfoHarness to provide keyword, attribute, and content-based access on textual, structure, and visual data.

Users can set up annotation relationships among artifacts or relationships correlating all documents about a certain subject without explicitly including them in a collection. Each IHO can also participate in multiple relationships and be part of multiple collections. This lets users impose more than one logical view on a given set of IHOs. For example, certain users might want to browse a company personnel repository by departments and groups, whereas others might want to browse the same

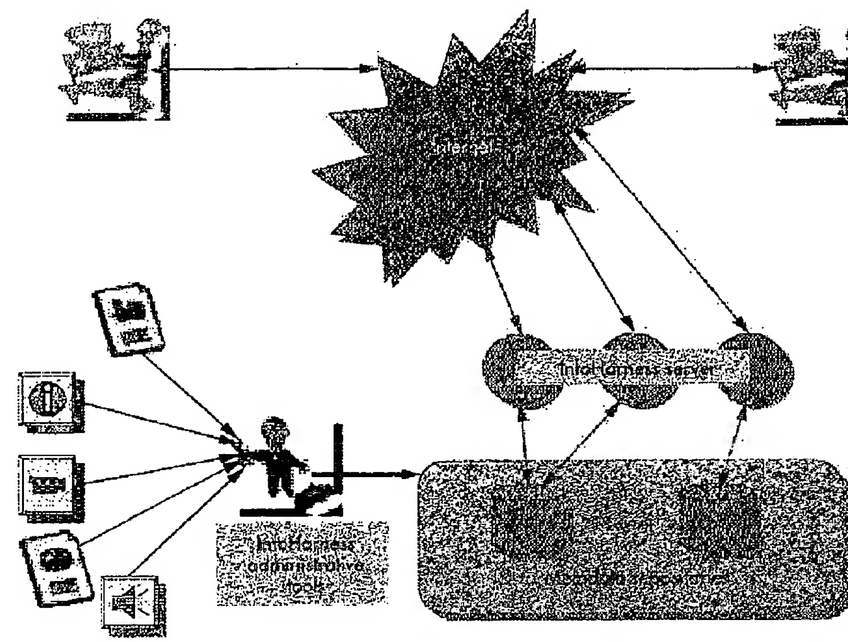

Figure 1. High-level InfoHarness architecture showing the three main system components.

repository by subject expertise. We could impose both these views—and collection structures—on the same repository.

An administrator can designate that a collection build a content index from the text of the collected documents. In this case, the extractors pull out the text body or a selected segment of the text from appropriate document types and pass them back to the administrative system. The system then uses the extracted text bodies to build the keyword index. Upon locating any of these text bodies through a keyword search, InfoHarness automatically refers the user to the IHO that encapsulates the information artifact associated with the text body. For such collections, the user can submit a content-based query, which uses the index to find documents that contain user-specified, keyword-based search criteria.

System Architecture

Figure 1 depicts a client-server system in which the clients are HTTP-compliant Web browsers and the servers are InfoHarness servers providing access to documents registered with a particular repository. When a user connects to an InfoHarness server, the system dynamically produces HTML pages that constitute the user interface. By activating links and using HTML forms, the user can navigate, search, and access the information maintained by that server.

In response to user requests, the server can invoke multiple associated methods for each IHO. For any document type, users can define type-specific methods for displaying the artifact associated with an IHO. The simplest method involves using an appropriate MIME type to display the original artifact on the client browser. In a more complex method, a user might click on an object, and InfoHarness could send the underlying artifact as an e-mail attachment. Other methods might translate the artifacts into HTML to allow the clients to display them inline. Methods also let users perform tasks such as sending a document to the printer or sending a facsimile of the document to another user.

A single IHO can be associated with multiple methods; the user can choose among them at runtime. For example, to display IHOs encapsulating Unix manual (man) pages, the user could choose the Unix utility xman or have InfoHarness translate the pages to HTML and display them within the Web client.

InfoHarness also has an extensive set of administrative tools for managing authorization and security.

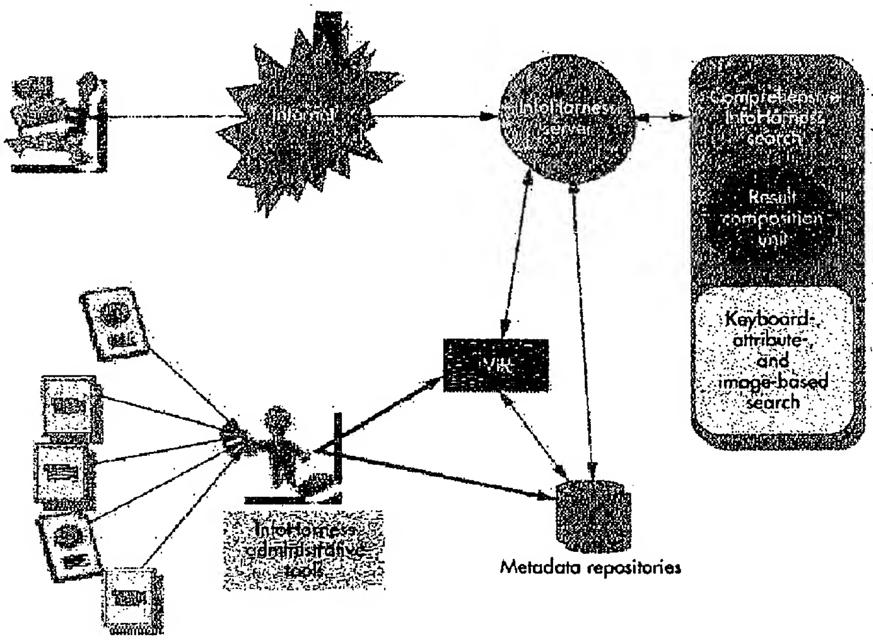

Figure 2. The VisualHarness architecture showing the use of VIR as a black box.

VISUALHARNESS: VISUAL INFORMATION MANAGEMENT

The VisualHarness system extends the basic InfoHarness to deal with visual data, giving users the ability to search and access distributed repositories of text, images, and other types of data, including structured databases. In addition to keyword and attribute-based queries, it also supports content-based queries over images, involving color, texture, composition, and structure. A VisualHarness metabase can consist of indices (for example, a full-text index for textual data and a feature-based index for image data) and attribute-value pairs. The attribute-value pairs support attribute- and content-based access of visual data using a novel approach that converts feature vectors into structured metadata. Like InfoHarness, VisualHarness has an open and extensible architecture that provides hooks for using various third-party indexing engines for textual data and visual information retrieval (VIR) engines, such as Virage's.[5]

Figure 2 shows a high-level view of the VisualHarness architecture. The InfoHarness server accepts a user query as a client request from a browser, and the query engine module of the query-processing unit (QPU) creates subrequests for the relevant search components. The search components use metadata (precomputed and stored in the metabase or computed at runtime) to determine references to the relevant data and then provide them to the QPU's result composition module. The QPU normalizes, rescales, and formats the result, which the InfoHarness server then displays to the user. When the user selects one or more data objects for display, the server directly accesses the appropriate repositories to retrieve the data.

The query-processing subsystem uses weighting strategies for a scalable approach. That is, a user can assign different weights to different kinds of similarity. The system then restricts database information retrieval according to the assigned weights. Suppose the VIR engine supports three properties—P_1, P_2, and P_3. The user can assign each of these properties different weights—i_1, i_2, and i_3—so that the VIR bases its retrieval on

$$i_1P_1 + i_2P_2 + i_3P_3, \text{ where } 0.0 \leq i_1, i_2, i_3 \leq 1.0.$$

The VIR normalizes and scales the resulting values to rank each of the objects retrieved. This access method applies if the system can access and understand feature vectors. Because VisualHarness has

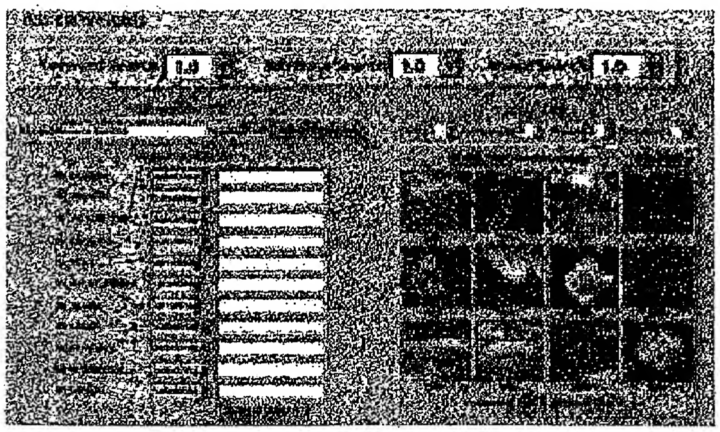

Figure 3. Comprehensive search screen in the VisualHarness system.

neither access to the actual feature vectors of an image object nor a way of interpreting them, it uses the VIR engine as a black box.

The Black-Box Approach for Content-Dependent Metadata

Feature vectors from an image refer to the features extracted from different topological spaces. To rank the similarity of objects to a given query object, a system requires distances between the objects and the input query object. The black-box approach compares objects based on their differences with a reference image.

If R is a reference image and $O_1, O_2, ..., O_n$ are the objects in the database, the feature distance (the distance between any two objects O_1 and O_2 in the feature space) equals the absolute value (Euclidean distance, denoted by *abs*) of the difference between each object compared with the reference image for a particular property. That is,

$$D(O_1,O_2) = abs[D(O_1,R) - D(O_2,R)].$$

Feature vectors of the object sequence in the database based on different properties of an image are mapped to a point in the feature space; a query with tolerance *e* becomes a sphere of radius *e*.

The black-box approach allows high scalability because it does not limit information retrieval to a particular VIR engine and its corresponding image database. Runtime computation is not expensive, because the system precomputes the distance between each object and the reference image for each of its properties and stores these values in a metabase. Runtime computation basically involves retrieving the appropriate results from the database by converting the user query image, Q, into a database query D(Q,R)—the distance between image Q and the reference image.

Without this approach, during runtime the system would have to sequentially compute the distance between the query image and each image object in the databases. The black-box approach also lets us use different weighting strategies to combine the distances obtained in comparing each object with the reference image in that topological space. Because we are using normalized distances, we can also combine features computed using different engines.

Our initial black-box strategy was to use a null image as the reference. We chose an entirely black or an entirely white image, hypothesizing that such an image has no specific features and hence no properties of its own, and we obtained quite decent results. However, we continued to seek a reference image that would yield results more accurate than those obtained using the VIR engine directly. In our second strategy, the reference image is the "centroid" of the feature space. Ideally, this object should be equidistant from all the other objects. Because such an object would be difficult to construct, we chose an existing object close to the ideal location in the feature space.

We also investigated improving results by semantically correlating various objects into semantic groups. Members of such a group would have some binding feature, and objects could belong to multiple semantic groups. That is, we could "thread" objects based on some predefined semantics. By semantically correlating the objects, we make an effort toward better understanding the intent of the query. We investigated both content semantics and context semantics. Grouping based on content semantics relies purely on statistical principles and can be mathematically formulated, whereas context-based grouping might be automated, manual, or knowledge-driven. For further discussion of the black-box approach and various strategies for selecting reference images, including quantitative evaluations, see our other work.[6]

Figure 3 shows an example of the comprehensive search screen. A user could adopt any of the three search strategies (keyword-, attribute-, content-based) or combine them using relative weights. For images, within the content-based search, the user could add color, structure, texture, and com-

position to the search criteria. Iterative refinements are also possible.

APPLICATION BUILDING

The InfoHarness platform supports a wide range of customizations, such as additional document formats and screen layouts. An organization could customize the user interface headers and footers, or it could build a periodically running script that looks for expired documents and notifies the authors or a responsible administrator by e-mail. In addition, InfoHarness hooks let users create entirely new applications that work with a document control system, use a new indexing technology, or integrate with a billing and ordering system.

The major steps in using InfoHarness to build an application are

- Model and create InfoHarness collections and relationships.
- Decide which documents are part of which collections.
- Set application parameters.
- Design the process by which the application keeps information up-to-date.
- Design screen templates and widgets, implementing extractors if necessary.

This job requires a thorough understanding of information content and flow within the application's scope. The application builder must understand how the end user wants to find and access information, keeping in mind special requirements such as authorization and security.

For those who wish to extend the predefined system's functionality, InfoHarness gives system integrators an API to add programs or scripts. System integrators can, for example, enhance the application with additional document types or actions on document types, or add support for a new indexing package. Because document type methods can be any arbitrary script or wrapper, many integration scenarios are possible. Specific extractors and runtime methods let the application builder integrate legacy data, systems, and applications. These legacy systems could be wrapped and encapsulated as IHOs to suit the particular Web-based application.

Now let's take a step back to examine the activities that are likely to be involved in building an application.

Information Inventory and Modeling

Taking inventory of the most valuable information can be as simple as consulting project documentation control or as difficult as creating a special-purpose committee for the task. Several items are useful at this stage:

- a draft inventory of all frequently used resources,
- a scenario describing expected information consumers and usage patterns,
- a list of entry points into the organization for new information, and
- a few friendly users who agree to provide feedback on the application.

Information modeling involves defining the structure through which users will find resources. This organization is important in the application's perceptual ease of use. We suggest these guidelines:

- *Depth versus width.* A primary decision will be how much information to put at any given level and how many levels to create. Deep applications need many specific collections, while broad applications need fewer general collections. The more clicks it takes for users to find what they are looking for, the more likely it is that they will end up in the wrong place.
- *Customer focus.* Information-modeling interests will conflict, but the most frequent and important users should expect the smoothest ride. If technical people are the intended primary users, then technical collections should be at the top level. If managers or customers are the primary users, technical details might be several levels down, available when necessary.
- *Frequency of use.* Make a collection of hot items that people refer to most frequently. This could contain, for example, project status reports, templates, or subject matter expert lists.
- *Indexes at appropriate levels.* Global indexes are useful for shot-in-the-dark searches, but they may not be focused enough for experts hunting for a specific detail. Consider, for example, a software-engineering repository that contains collections for designs, code segments, requirements, test cases, and so on. A developer searching for reusable components will be better off searching an index of the code collection rather than the entire repository.
- *Learning by example.* Survey related information systems and study their organization. Many information applications on the Internet illustrate good ways to model data.

Once you have drafted the collection structure, you might prototype a subset of the application and have friendly users provide feedback.

Resource Translation

Although InfoHarness includes a wide variety of ready-to-use extractors, these cannot cover every conceivable application need. Writing new extractors can be trivial, provided you have the necessary technology to parse the given document type. Then, you simply plug in the new extractor. Recompiling the InfoHarness server is not necessary, although you may have to recompile the extractor if it is not written in a scripting language.

The logical application model need not have any relation to the physical distribution of the original documents. Once you set up the logical model, InfoHarness completely shields the user from the underlying physical structure. In rare cases, the application builder might move data to a more central location or convert it to a single format. This makes building an application simpler, but InfoHarness can bring together widely distributed information as well.

Interface Design

Users usually require help or additional details. Place necessary guidelines, help screens, hypertext links, method launch links, and contact information in strategic places. For example, the server uses screen templates or widgets to dynamically generate the search screens; the application builder could customize these templates to the users' needs.

Initial Repository Creation

Bring the application resources (or the metadata) into a single place, possibly with searchable indexes for some or all sets of data. This involves using a set of administrative tools to build up a set of objects and relationships in the metabase. Use the InfoHarness tool set to write higher level scripts that define the repository structure and determine how collections are built.

Repository Maintenance Mechanism

When an application demands timely update of resources, establish automated procedures to find new resources, translate them if desired, and add them to the repository in the correct groupings. Maintenance scripts exist, but these do not automatically update resources or find new resources. You will have to write your own daemon (or agent) for that.

System Integration

If you are integrating one or more existing systems or interfaces, you must supply specific wrappers. InfoHarness has a well-defined flexible interface, which can bring different systems together. Consider a personnel application that has its original artifacts in a legacy system. The system integrator has access to scripts that can run as clients on the Unix side and that execute queries on the mainframe back end. The extractors could be wrappers for these Unix scripts.

Suppose an IHO corresponds to an employee: Given an employee identification number, the extractor wrapper could invoke the existing scripts and return metadata from the legacy system that, after reformatting, would be encapsulated and inserted into the metabase. At runtime, a similar wrapper would invoke the existing scripts to display employee information to the user. Employees could be included in multiple collections based on group, department, or other factors.

Client Configuration

Setting up the Web browsers with the proper viewing utilities can be a major consideration, depending on which data formats the application will use. Unix machines cannot view PC-created documents without considerable translation or sophisticated tools. InfoHarness cannot directly aid in this phase, but its support of extensible methods can make the task simpler.

The InfoHarness commercial version (see the sidebar, "Related Work") has a PC patch called PC-AdaptX, which provides extractors for Microsoft Office and other documents. However, PC-AdaptX uses the InfoHarness utilities provided on Unix, so you should have a PC network file system (PC-NFS) set up to cross-mount your PC file system on your Unix platform.

ONGOING WORK

This article does not cover all of our work on InfoHarness. One InfoHarness extension provides access to relational databases. This lets users browse and have keyword- and attribute-based access to collections of objects defined over relational tables from multiple DBMSs. The extended system also supports audio, video, 3D, and other spatial data. Other recent work on InfoHarness has extended the query subsystem to let users combine attribute, keyword, and image feature searches over InfoHarness repositories. We also researched the issue of scalability by allowing access to and combining

RELATED WORK

Fulcrum—an example of system that serves mainly as a full-text-indexing engine (see http://www.fulcrum.com/), does text and attribute searches and has an extensive library for different file types. However, unlike InfoHarness, it requires all non-HTML documents to be converted to HTML. Fulcrum's structured attribute-searching capability resembles the InfoHarness metabase from an interface perspective, but Fulcrum treats the attributes as tags associated with the document, whereas InfoHarness provides a more object-oriented view. Fulcrum does not attempt to provide an integration platform for heterogeneous data. Another similar commercial platform is Verity (see http://www.verity.com/).

Many commercial DBMS vendors have integrated full-text searches with their engines, as Oracle has with ConText. ConText enables full-text indexes over text bodies in the DBMS. Informix Universal Server provides text DataBlades that enable full-text searches over large text bodies. These approaches allow keyword and attribute searching but require the user to import the text into the DBMS.

Harvest is an integrated set of tools that gather and distribute indexing information, support the easy construction of many different types of indexes customized to particular information collections, and provide caching and replication support to alleviate bottlenecks. Harvest also includes a distributed set of components for extraction and indexing. Like InfoHarness, it supports metadata extraction and indexing/searching, but it doesn't allow construction of logical browsing models. The Netscape Catalog Server is very similar to Harvest and shares the same architecture.

Rufus uses an object-oriented database to store descriptive information about a user's data and a full-text retrieval database to provide access to the data's textual content (see http://www.almaden.ibm.com/cs/showtell/rufus/overview.html). Rufus uses extractors to pull out metadata and builds browsable structures similar to InfoHarness collections. It has a good type-recognizer system, but it is not very extensible. Unlike InfoHarness, Rufus does not support dynamic extensions to the object schemas. In addition, Rufus does not allow users to plug in third-party indexers, and it uses a proprietary information retrieval system.

The Intermedia system, based on the encapsulation of information units, allows many-to-many relationship linking and composite objects (see http://[illegible]/intermedia.html). This system's primary purpose is heterogeneous document management, and it shares many of the concepts used in InfoHarness. It is still an evolving prototype.

Hyper-G/Hyperwave combines the DBMS approach with open Web technology (see http://www.hyperwave.de/). This system offers dynamic hyperlink management, hyperlinks from and to arbitrary documents, integrated text and attribute indexes, and integrated user rights and session management. However, this system does not have an extensive metadata extraction system. Type support is limited and not extensible. Also, it restricts the object schema to a predefined set of attributes and uses a proprietary indexing system. The user interface is quite dynamic and fully configurable.

Finally, several existing systems[1-3] manage metadata about large, diverse sets of data, such as those from satellite, environmental, and weather studies. These systems have tools for metadata extraction, but they are usually based upon information extraction from DBMS objects like tables, schemas, transactions, and data values. They address a range of problems very different from those that InfoHarness addresses.

InfoHarness has been used as a platform for two research projects—see http://lsdis.cs.uga.edu/proj/iq/iq.html and http://lsdis.cs.uga.edu/proj/ih/ih_pub.html. The system's commercial version is called Adapt/X Harness.

REFERENCES

1. L. Mark and N. Roussopoulos, "Metadata Management," Computer, Vol. 19, No. 12, Dec. 1986, pp. 26-36.
2. S. Chakravarthy et al., "A Metadata Management System to Support Data Interoperability, Reuse, and Sharing," J. Database Management, Vol. 5, No. 2, 1994, pp. 30-40.
3. L. Shklar et al., "New Approaches to Cataloging, Querying, and Browsing Geospatial Metadata," Proc. Second IEEE Metadata Conf., IEEE Press, Piscataway, N.J., 1997.

results from multiple data partitions, each with its own independent and possibly different indexer. We have also made infrastructure layer extensions to allow the use of CORBA for distributing the InfoHarness repositories themselves.

Taking the second-generation InfoHarness concepts[7] a step further leads us to the InfoQuilt system,[8] a third generation of information interoperability and integration that we are currently at work on. InfoQuilt's purpose is to provide intelligent analysis, mining, fusion, and dissemination of heterogeneous media data. Its architecture has three layers—data, metadata, and user/domain models or ontologies. With extensive support for metadata and user models, InfoQuilt will define context, support user profiles and models, use media-inde-

pendent correlation to represent media-independent semantic relationships,[9] and eventually support multiple domain-specific ontologies.[10] InfoQuilt will allow traditional keyword-based queries as well as high-level information requests involving attribute-based, iconic, mixed-media, and concept-based information requests. This will involve distributed management and correlation of heterogeneous media stored at different locations. The system will also support information analysis and fusion at a higher semantic level with the help of human-directed browsing, searching, accessing, and correlation of heterogeneous media data. ■

ACKNOWLEDGMENTS

InfoHarness is a trademark of Telcordia Technologies Inc. Partial support for this research came from the Massive Digital Data Systems program. We thank Satish Thatte, the product manager for the initial prototype and the commercial versions; Leon Shklar for the first prototype; the InfoHarness team (Bret Gorsline, Paul Hughes, Steve Ince, Vipul Kashyap, Tom Mok, Bob Mowry, and Andy Werth); and the VisualHarness team (Srilekha Mudumbai and Krishnan Parasuraman). Tarcisio Lima provided extensive help in editing this article to its current length. We also thank Virage and Informix for software donations to the VisualHarness project.

REFERENCES

1. A. Sheth and W. Klas, eds., *Multimedia Data Managing, Using Metadata to Integrate and Apply Digital Media*, McGraw-Hill, New York, 1998.
2. U. Fayyad et al., eds., *Advances in Knowledge Discovery and Data Mining*, MIT Press, Cambridge, Mass., 1996.
3. S. Deerwester et al., "Indexing by Latent Semantic Indexing," *J. Am. Soc. Information Science*, Vol. 41, No. 6, 1990.
4. B. Kahle and A. Medlar, "An Information System for Corporate Users: Wide Area Information Servers," *Connexions—The Interoperability Report*, Vol. 5, No. 11, Nov. 1991.
5. A. Gupta, *Visual Information Retrieval: A Virage Perspective*, tech. report, Virage, San Mateo, Calif., 1995.
6. A. Sheth et al., "Searching Distributed and Heterogeneous Digital Media: The VisualHarness Approach," *Proc. IFIP 2.6 Working Conf. Database Semantics*, North-Holland, Amsterdam, 1999, pp. 311-330.
7. A. Sheth, "Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics," *Interoperating Geographic Information Systems*, M.F. Goodchild et al., eds., Kluwer Academic, Boston, 1999.
8. V. Kashyap and A. Sheth, "Semantic Heterogeneity in Global Information Systems: The Role of Metadata, Context and Ontologies," *Cooperative Information Systems: Current Trends and Directions*, M. Papazoglou and G. Schlageter, eds., Academic, 1997, pp. 139-178.
9. K. Shah and A. Sheth, "Logical Information Modeling of Web-Accessible Heterogeneous Digital Assets," *Proc. Forum on Research and Technology Advances in Digital Libraries*, IEEE Computer Soc. Press, Los Alamitos, Calif., 1998, pp. 266-275.
10. E. Mena et al., "OBSERVER: An Approach for Query Processing in Global Information Systems Based on Interoperation Across Pre-existing Ontologies," *Proc. First IFCIS Int'l Conf. Cooperative Information Systems*, IEEE CS Press, Los Alamitos, Calif., 1996, pp. 14-25.

ADDITIONAL READING

- M. Carey et al., "Towards Heterogeneous Multimedia Information Systems: The Garlic Approach," *Proc. RIDE-DOM*, IEEE CS Press, Los Alamitos, Calif., 1995, pp. 124-131.
- V. Kashyap and A. Sheth, "Schematic and Semantic Similarities between Database Objects: A Context-based Approach," *The VLDB J.*, Vol. 5, No. 4, 1996, pp. 276-304.
- J. McCarthy, "Metadata Management for Large Statistical Databases," *Proc. Eighth Int'l Conf. Very Large Data Bases*, IEEE Press, Piscataway, N.J., 1992, pp. 234-243.
- L. Shklar et al., "InfoHarness: Use of Automatically Generated Metadata for Search and Retrieval of Heterogeneous Information," *Proc. CAiSE-95*, Springer-Verlag, Berlin, 1995.

Kshitij Shah is currently a senior consultant at BEA Systems. From 1994 to 1996, he was member of the technical staff at Bellcore, where he served as chief architect and developer of AdaptX/Harness. In 1997, he worked at the LSDIS lab as a research assistant professor in the Department of Computer Science at the University of Georgia. In 1998, he served as the senior solutions architect and senior project manager at SoftPlus Inc. Shah received an MS in computer science from Rutgers University.

Amit Sheth is a professor of computer science and director of the Large-Scale Distributed Information Systems Lab at the University of Georgia. Previously, he worked in R&D groups at Honeywell, Unisys, and Bellcore. Several research efforts he has initiated and led have resulted in commercial usage or products, in part through the companies he founded—Infocosm Inc. (http://www.infocosm.com) and Taalee Inc. His interests include semantic interoperability and integration involving digital media, information brokering, and enterprise-wide and multiorganizational processes. Sheth received a BE from BITS, Pilani, India, and an MS and a PhD from Ohio State University.

Readers may write to Kshitij Shah at BEA Systems WebXpress Division, 2315 North First St., San Jose, CA 95131; kjshah@beasys.com; http://www.beasys.com/. Amit Sheth is at the Large-Scale Distributed Information Systems Lab, University of Georgia, Athens, GA 30602-7404; amit@cs.uga.edu; http://lsdis.cs.uga.edu.